AMENDMENTS TO THE SPECIFICATION:

Please replace paragraph [0052] on page 14 with the following:

[0052] FIG. 3 is a flow chart showing the use of the manufacturer certificate 36 in a secure boot loader 50 and a secure boot checker program 52, preferably stored in ROM 22 to protect the programs from alteration of program flows. The secure boot loader determines whether boot system firmware is available for uploading at power-up. If so, the secure boot loader first loads a flash programmer. The flash programmer is used to load the system boot firmware. The flash programmer must also have a manufacturer certificate 36 and the secure boot loader is responsible for ensuring the authenticity and integrity of the flash programmer's manufacturer certificate and the code of the flash programmer program prior to any execution of the flash programmer. The flash programmer then uploads the system boot firmware.

Please replace paragraph [0057] on page 15 with the following:

[0057] In an alternative embodiment, a hashed value for the manufacturer's public key is stored in manufacturer certificate 36; in this case, hashing step 100 can be eliminated. Also, only a predetermined number of least significant bits of the hashed manufacturer's public key can be stored in the eFuse memory ~~1~~24; in this case, only corresponding bits would be compared in step 104.

Please replace paragraph [0099] on page 24 with the following:

[0099] The platform certificate 38 makes use of the KEK stored in eFuse memory ~~1~~24. In the preferred embodiment, the KEK is a random number generated on-chip during production, such that the value of the KEK is not known to anyone. The KEK in the eFuse memory ~~1~~24 such that it is not accessible through I/O ports or to application software. It is desirable that each chip's KEK be used in a manner that it cannot be externally determined or intercepted by other programs. While storage of the KEK in the eFuse memory ~~1~~24 allows determination through physical observation of the fuses in the fused memory, such observation can only be made upon destruction of the chip itself; since each chip generates its own KEK, knowledge of one chip's KEK will not compromise the security of other chips.

Please replace paragraph [0100] on page 24 with the following:

[0100] The KEK is used to encrypt other software keys that are randomly generated during operation of the device. As shown in FIG. ~~9~~10, a random number-generator 250 (which could be either a hardware or software implementation) generates a random software key (SW_KEY) as necessary. Hence, each application may be associated with a different software key. SW_KEY is encrypted using the KEK in step 252 and stored in the platform certificate 38 as ENC_SW KEY. Since ENC_SW_KEY can only be decrypted using the KEK, and since the KEK is secret and internal to the chip, ENC_SW_KEY can only be decrypted to applications that have access to the KEK. Thus, only the system software in ROM should have access to the KEK.

Please replace paragraph [0120] on page 30 with the following:

[0120] FIG. 15 illustrates an alternative design for accessing the device 10 is a certain mode, such as a test mode shown in FIG. 15. This design stores the hash of an access code (H_Test_ID). This code could be stored in the eFuse memory 24. To access the test mode, the party would need to enter an access code (Input_Test_ID). Input_Test_ID is hashed in block 330 and compared to H_Test_ID in block 332. If the hashed access code from block 330 matches the stored hashed access code, then entry to the mode is enabled.